EXHIBIT 76

Dynamic Revenue Sharing (DRS) V2 Proposal

mirrokni@,renatoppl@
in collaboration with whlin@, jimgiles@, nitish@, mpal@, gagangoel@

Background and Summary

Dynamic revenue sharing (DRS) for DRX is a new way of changing revenue share per auction and as a result, increase the fill-rate and total revenue of publishers in DRX. Several variants have been proposed for this purpose. Here, after comparing with several variants, we propose a "revised 2sided DRS" (the idea is summarized below) as the DRS V2. This revenue sharing mechanism encourages seller-friendly bidding from buyers by encouraging them to declare two bids from their advertisers. More notably, compared to DRS V1 and other variants, it

- brings more revenue lift for publishers (even while charging the reserve price instead of first price in the dynamic region),
- keeps AdX's margin at 20% (& thus it brings (much) more profit lift to AdX),
- more explicitly takes into account incentive issues from both buyer and seller sides across auctions, and results in much less opportunity to game the auction by bid shading (compared to V1 and other non-truthful variants),
- strictly increases ROI for seller-friendly buyers (e.g., GDN) by increasing their utility per each auction,
- treats seller-friendly buyers (e.g., GDN) the same as DRS V1 (in terms of pricing & allocation),
- finally, it encourages declaring two bids for the exchanges.

In this document, we present a detailed comparison to other DRS methods and show the above points. Most notably, our experiments show that this algorithm not only brings more revenue for AdX and publishers, but also it increases RPM for publishers a bit. At the same time, its impact on the advertiser RPM and publisher RPM is not significant. Here, we present this DRS method, report its comparison to V1 and other truthful and non-truthful variants.

Compared to DRS V1, we get considerable more profit (at least 53% profit lift with a conservative estimate). Finally, we note that by implementing DRS V2, we can apply a less tight throttling probability for DRS V2 compared to DRS V1, and therefore we can get much more revenue (>55% more revenue lift, and > 169% more profit lift).

Links to previous proposals: DRS V1, Enhanced DRS, Two-sided DRS, Truthful DRS

Description of the Revenue Sharing Scheme

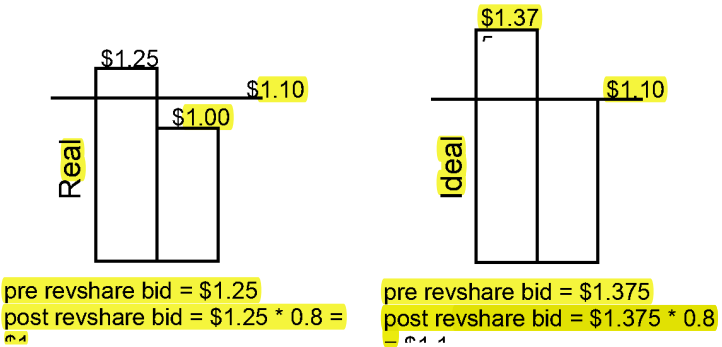
The main goal with DRS is to be able to clear impressions for which the highest bid is very close to the reserve. Currently sellside revshare[1] is fixed at 80% and the computation of revenue (how

[1] there is also buyside revshare. The bid submitted by the advertiser is multiplied by buyside revshare (which can be smaller or larger than 1.0 due to Bernanke) and then multiplied by sellside revshare

much we charge the advertiser) and payout (how much we pay the publisher) is done as follows:

- post_revshare_bid = bid * revshare (which map bids to publisher space)
- if no post_revshare_bid above the publisher reserve, the auction doesn't clear.
- o.w. payout = max(reserve, second highest post_revshare_bid)
  revenue = payout / revshare

So a bid of $1.25 won't win an impression with reserve $1.10, since first we take 80% of sellside revshare, so the post-revshare bid is $1 and the impression doesn't clear. See example below



The two-sided dynamic revshare proposal is to clear impressions that would clear if it were not for revshare, i.e., impressions for which **bid * revshare < reserve < bid**. We say say such impressions are in the ***dynamic region***.

In the two-sided design, whenever an impression is in the dynamic regions, we clear it and charge:

- revenue = max(reserve, second highest bid)
- payout = reserve

In this query, the publisher has up to 100% of the revshare. So in order to recover Google's 20% cut on this transaction we have for each buyer and for each publisher a **debt account** in which we store the amounts we should have been paid in such queries:

- debt [buyer] += reserve / revshare - revenue
- debt [publisher] += payout - revenue * revshare

We attempt to collect debt in a later query for which bid * revshare > reserve, which we call the **non-dynamic region**. In such query we do as follows:

- $payout^0$ = max(reserve, second highest post_revshare_bid)
- $revenue^0$ = $payout^0$ / revshare
- revenue = $revenue^0$ + buyer_collection

(currently constant at 80%). In this doc, we assume buyside revshare is always 1.0 (or equivalently, when we say pre-revshare, we actually mean, post-buyside-pre-sellside revshare.

- payout = revenue * revshare - publisher_collection

and then we decrease their credit by the amount collected:

- debt [buyer] -= buyer_collection
- debt [publisher] -= publisher_collection

Where we are careful to pick buyer_collection and publisher_collection to preserve constraints such as:

- don't increase the price too much from the non-DRS outcome, so we do:
  buyer_collection < alpha * revenue^0
- don't take more than a fraction of the bid price gap:
  buyer_collection < beta * (bid - revenue^0)
- pay to the publisher at least his reserve:
  publisher_collection < revenue * revshare - reserve
- and of course, not collect more than the current debt:
  buyer_collection < debt [buyer]
  publisher_collection < debt [publisher]

Comment [1]: Why not allow a small collection greater than current debt? (That is, allow debt to be slightly negative, and if the buyer / publisher ends up with negative debt, we refund the money.)

Finally we consider a second way in which buyers can decrease their debt (equivalently: acquire credit) whenever they declare a 2nd bid which is higher than the max(2nd other bid, reserve) by declaring a min_cpm_payment. We do:

- debt [buyer] -= min_cpm_payment - max(2nd other bid, reserve)

This way, we give explicit incentives for buyers (not only GDN) to declare more than one bid, and as a byproduct, the revised 2sided DRS never increases the price set above max(2nd bid, reserve). This scheme naturally incorporates the incentives of buyers who play in favor of the publisher and the exchange and does not increase the price for these buyers.

Throttling Schemes

The dynamic revshare scheme presented above heavily relies on buyers bidding out of the in the dynamic regions so that we can collect debt back. In order to prevent buyers from only bidding in the dynamic regions, we implement throttling schemes, which prevents a buyer from being too often in the dynamic region:

- probabilistic throttling: given a target revshare (say 19%) we compute probabilities p for buyers and sellers such that they are eligible to enter the dynamic region with probability p. With probability 1-p, the auction doesn't clear.
- threshold throttling: we compute thresholds such that we allow a certain query to be cleared in the dynamic regions only if the revshare obtained from that particular query is at least the threshold.

Impact on the Eco-system

AdX is in competition with other exchanges, and both publishers and buyers have an option to go to other exchanges. The exchange should care about both sides of the market and provide more features to both sides to thrive in face of competition with other exchanges. Fortunately,

the proposed dynamic revenue sharing system benefits publishers and advertiser due to more efficient allocation and pricing. The impact on the publisher payout and the exchange is significantly positive while the impact on the buyers is almost positive. In the current market, increasing pay-out and keeping high RPM for publishers are critical factors and as summarized above, the impact in all these directions are positive.

Point-wise improvement for seller-friendly buyers (e.g., GDN)

The proposed DRS V2 implicitly encourages buyers to provide two bids (in other words, if they submit min_payment_cpm). Such a buyer (e.g., GDN) is seller-friendly in that it is trying to benefit the whole eco-system by providing more value to the sellers (or publishers). When compared to fixed sellside revenue share, a seller-friendly buyer (which submits min_payment_cpm) is better-off for *each particular auction* in DRS v2: if an auction was won at price p, then by submitting the same bid and facing the same competition, the auction will also be cleared at price p. Besides that, other impressions which previously had a certain price, will be available to the buyer at a lower price. Only for those impressions which weren't available before, the price paid won't be equal to the threshold price. In particular, this means that Bernanke simulations will be better off in the DRS world.

Comparison to Alternatives for Revenue & Publisher/Buyer RPM

We compare the different revenue sharing schemes in terms of their revenue in blue (total amount collected to the buyer) and profit in red (total amount collected from buyers minus amount paid to the seller).

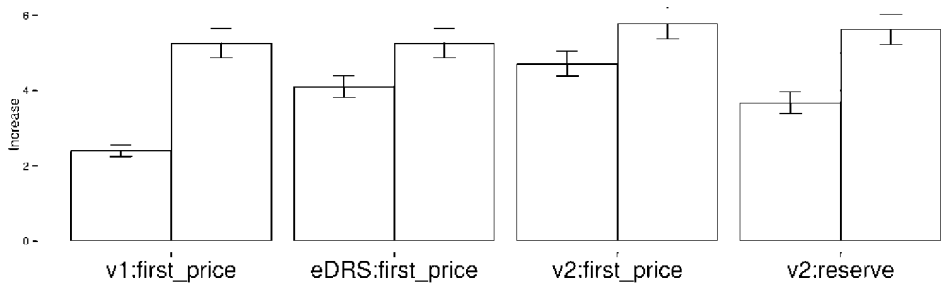
We consider V1, using first price, then enhanced DRS (which only collects debt from buyers) and version 2. Finally, we consider version 2 with reserve[2].

Comment [2]: version 2 here is what we called revised. To avoid confusion, I have only one version 2, without revised vc non-revised.

In the following plots, we compare the revenue and profit obtained from RTB buyers in each scheme compared to the revenue and profit obtained from them if DRS is completely disabled. The following graphs don't include the GDN contribution. The lift obtained from GDN could not be evaluated since GDN doesn't bid below the stated reserve price.

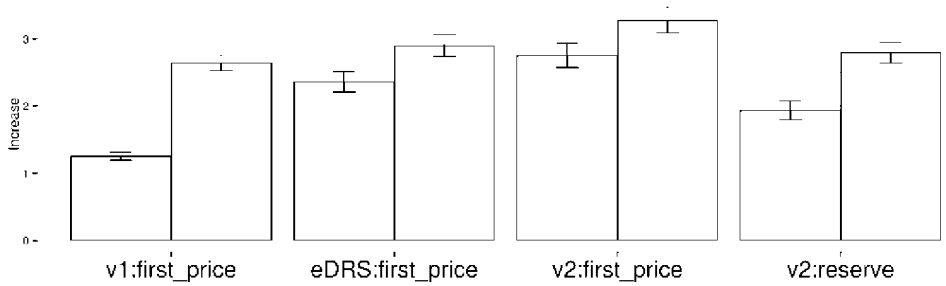
The following represent profit and revenue lift for the unthrottled DRS schemes:

[2] In the simulations we place a restriction that v2 is only allowed to increase the original price by at most 10% outside the dynamic region and take at most 50% of the bid-price gap.



experiment_id	profit lift	revenue lift
v1:first_price	2.40%	5.26%
eDRS:first_price	4.10%	5.26%
v2:first_price	4.71%	5.78%
v2:reserve	3.68%	5.63%

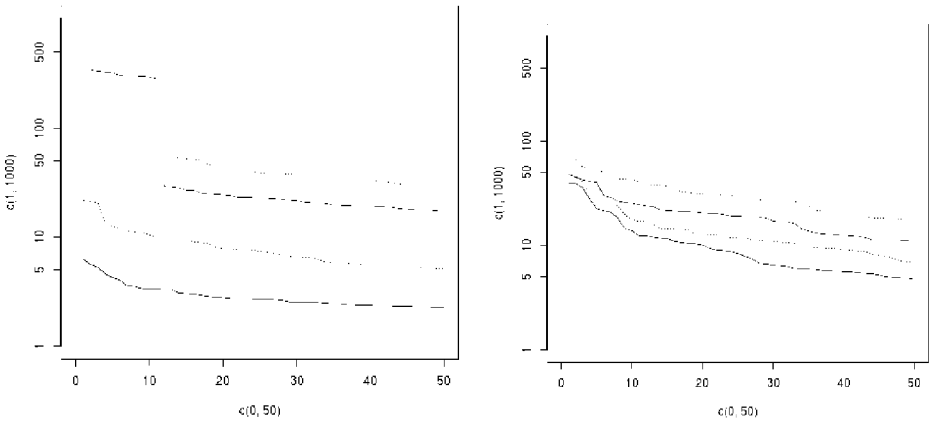
Note that eventhough eDRS achieves a good increase in profit lift, it provides this increase in profit lift at the cost of the publisher payout lift. With 19% throttling we obtain the following:



In both cases, we note that the revenue is similar for different revshare schemes, but the profits are very different. In particular, v2:reserve produces a 53% higher profit lift then v1:first_price, even considering that v1 is clearing at the first price while v2 is clearing at the reserve.

Debt Distribution at the end

The amount of uncollected debt in the end can be seen as a measure on how untruthful the system is, since it corresponds to how much a buyer or publisher can 'game' the system. The point (k, debt) corresponds to the debt of the buyer/publisher with k largest debt.



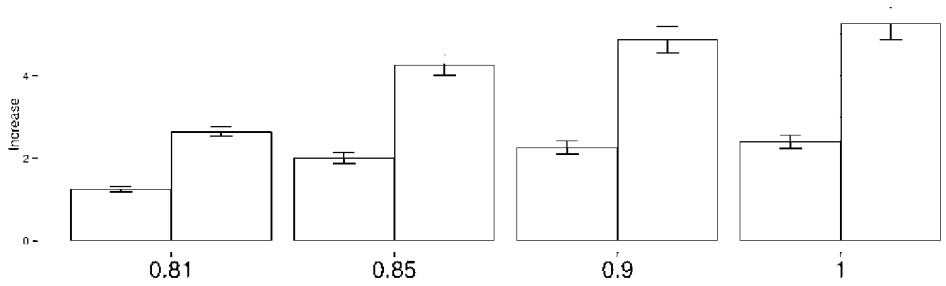
k-th larger debt (buyers) k-th larger debt (publishers)

(Red = DRS v1, Green = enhanced DRS; Blue = DRS v2)

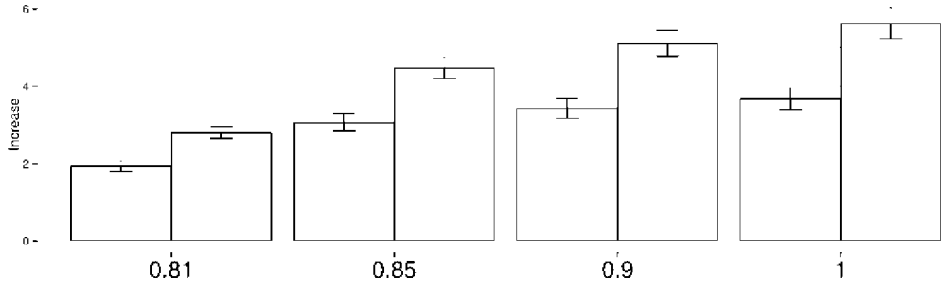
(fullline = 19% throttling, dotted line = 15% throttling)

Impact of Throttling Rate

Below we plot revenue lift (blue) and profit lift (red) as a function of the throttling rate. Throttling rate r means that a buyer or seller is throttled if his revshare exceeds r. So, throttling rate r = 1.0 means unthrottled.



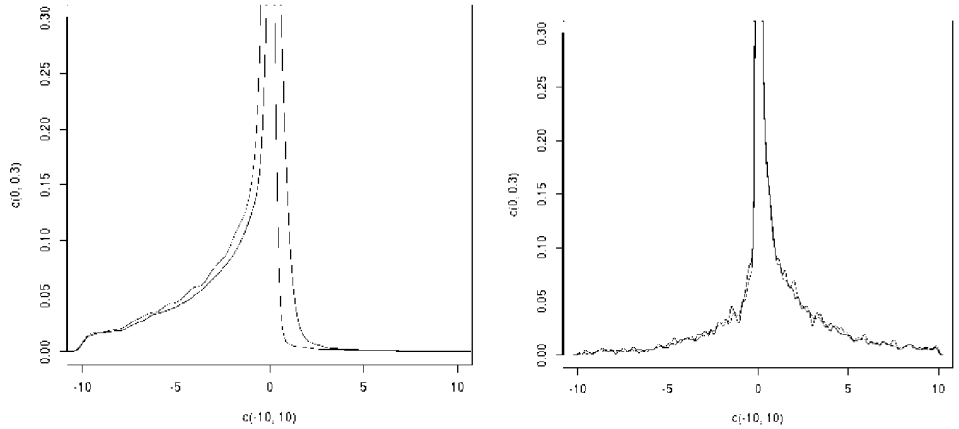
DRS v1



DRS v2

RPM impact on Buyers and Sellers

Since DRS clears impressions with lower reserve, it tends to decrease RPM for publishers (where RPM here is the ratio between payout to publishers and volume of impressions). The following plots show that RPM slightly decrease for buyers (but mostly the decrease is less 5%). For buyers it is distributed in a more or less symmetric way around zero.



Change in RPM for sellers Change in RPM for buyers

(Red = DRS v1, Blue = DRS v2)

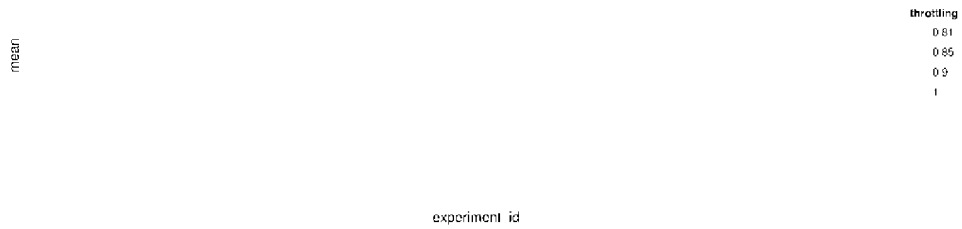
The x-axis represents a change in RPM (0.1 = 10%) and the y-axis represents the density, i.e., the y-axis is proportional to the # of buyers/sellers that experienced that change in RPM.

GDN Case Study: Credit vs. Debt accumulation

In revised DRS, the total amount of credit awarded to GDN for submitting a min_cpm_payment amounted to 70 times more than the debt produced by the same buyer. This means in particular that with current bids, GDN wouldn't accumulate any debt, since all of it would have been offset by awarded credit.

Final RevShare Distributions

The following plot shows the final overall revshare obtained by each DRS scheme (in the x-axis). Each curve corresponds to a different throttling rate used. In particular, DRS v2 with 15% throttling generates the same overall revshare as DRS v1 with 19% throttling.



These final revenue shares are also correlated with opportunity to game the auction by bid shading (since the total discount the buyer gets is proportional to this value). The above curves also imply that DRS V2 provides much less opportunity to game the auction by bid shading.

From the above charts, we note that by implementing DRS V2, we can apply a less tight throttling probability for DRS V2 compared to DRS V1. Now by comparing the revenue lift and profit lift numbers for 19% and 15% throttling probability listed previously, we conclude that we can get >55% more revenue lift, and > 169% more profit lift from DRS V2 compared to DRS V1.

Incentive-friendliness across auctions

The two sided version is incentive-friendly in two ways:

- since it clears queries on the reserve (and not at the first price), this scheme doesn't create much bid-price correlation.
- it is (almost) incentive-compatible in aggregate: if a buyer bids b in the dynamic region and hid debt is collected back later, is as if he had originally bid reserve / revshare in that particular query. So it is as if he participates in a plain second price auction with the contract that 'whenever my bid is between reserve and reserve / revshare, I agree to bump it up to reserve / revshare'. Note that the best strategy of the buyer is simply not to

Comment [3]: Assuming quasi-linear utility? Make this explicit, because buyers may like more buyer revenue, not more buyer profit. (Also, they may be using first-price bids.)

bid in the dynamic region, in which case we recover the current state of the system. If they decide to bid, revenue and profit can only improve.

Comparison to per-auction truthful DRS

While the proposed DRS V2 takes into account incentives from buyers and sellers across several auctions, it does not implement a truthful auction per query, i.e., a buyer can benefit from changing its bid (e.g., by decreasing its bid) for one auction, but it will have to pay back the discount it has received later. Here, we note that there exists a truthful revenue sharing scheme that ensures truthfulness per auction (proposed by gagangoel@). Running similar simulations for this variant shows that the total revenue lift is much less for the truthful variant, and the total publisher payout may even decrease (although it's not significant). The results for this variant is summarized in this document.

Sensitivity to changing bids per auction

We note that with DRS V2 (or DRS V1), the price paid per auction may change slightly by increasing or decreasing one's bid. However, if a buyer tries to game the system and declare a lower bid, it has to pay above in other auctions. In other words, because of Note that for seller-friendly buyers like GDN, when the bid is in the dynamic region, this version charges the reserve price, and when the bid is not in the dynamic region, the auction charges reserve price divided by the seller side revshare. So the auction has the following properties:

- outside of the dynamic region, any bid larger than the price paid wins the auction by the same price paid.
- in the dynamic region, any bid between the bid and the price win the auction by the same price.
- in the dynamic region, any bid lower than the price paid loses the auction,

So at any point, there is one direction in which the threshold property is preserved. Moreover, we can make the pricing rule completely transparent and declare when the price charged was the reserve or the reserve divided by revshare.